الباب الثانى

أدوات تحكم الويب

٣ . استخدام نماذج الويب.

٤ . أدوات تحكم الويب.

٥ . المزيد عن أدوات الويب.

٦ . أدوات التحقق من صحة المدخلات.

الفصل الثالث
استخدام نماذج الويب

يحتوى نطاق **.NET** على نموذج برمجى جديد لتصميم صفحات الويب باستخدام **ASP.NET**. سنقوم فى هذا الفصل بالتعرف على السمات الأساسية لنماذج الويب **Web Forms**.

بانتهاء هذا الفصل، ستتعرف على:

- مفهوم نماذج الويب.
- مفهوم أدوات الخادم.
- إيداع البيانات والجولات الدائرية.
- كتابة كود نموذج الويب.
- أحداث أدوات التحكم.
- كتابة كود العميل **Client-side Script**.
- مثال تطبيقى.

ذكرنا فيما سبق أن هناك ثلاثة أنواع أساسية من التطبيقات داخل Visual Studio 2008 منها التطبيق النوافذى المعتاد لك من دراستك للإصدارات السابقة من Visual Studio إلى جانب نوعين جديدين هما تطبيقات الويب Web Applications وخدمات الويب Web Services وكلاهما مبنى داخل ASP.NET ويحتوى على الكود القابل للتنفيذ من قِبَل خادم الويب. كما تحتوى تطبيقات الويب على نموذج برمجى جديد يسمى نماذج الويب Web Forms الذى يمكنك من تطوير هذه التطبيقات بسهولة منقطعة النظير.

وعامةً يتكون تطبيق الويب من مجموعة من الملفات المرتبطة الموجودة داخل دليل تخيلى واحد، حيث يمكنك كتابة كود هذه الملفات بنفسك من خلال أحد محررات النصوص أو من خلال بيئة تطوير Visual Studio 2008. وبغض النظر عن الطريقة المستخدمة فى تطوير هذه التطبيقات، فإنك تستخدم ASP.NET ومن ثَمَّ استخدام النموذج البرمجى الجديد المعروف باسم نماذج الويب أو Web Forms. سنقوم فى هذا الفصل بالتعرف على هذا النموذج البرمجى الجديد، فكن معنا.

مفهوم نماذج الويب

حاولت مايكروسوفت جاهدةً الوصول إلى النموذج البرمجى لنماذج الويب لمضاهاة تطبيقات الويب بسابقتها الخاصة بالنوافذ أو ما يطلق عليه التطبيقات النوافذية. فالطريقة التقليدية فى برمجة صفحات الويب تتمثل فى كتابة الكود على خادم الويب الذى يقوم بدوره بتوليد كود HTML المكافئ وإرساله إلى مستعرض الويب الخاص بالمستخدم وهى الطريقة التى قمنا باستخدامها فى الفصل الأول حينما أنشأنا التطبيق العملى الخاص بألبوم صور الكتب. وهذه الطريقة على الرغم من عملها بكفاءة عالية إلا أنها غير أنيقة بالمرة. لبيان سبب ذلك، دعنا ننظر إلى الكود المطلوب لتمكين المستخدم من تحديث عنوانه من خلال إحدى صفحات الويب. فى تطبيق الويب المعتاد، يتم إجراء ما يلى:

١. يجب أن تقوم بالوصول إلى قاعدة البيانات وإظهار نموذج HTML للمستخدم.

ولأداء ذلك، استخدم ADO.NET (كما سنرى فيما بعد) للحصول على عنوان المستخدم من قاعدة البيانات وتخزينه داخل أحد المتغيرات ثم قم بعد ذلك من خلال الكود بإنشاء نموذج HTML مؤقت عن طريق تخصيص حقل العنوان بعبارة Response.Write كما يلى:

```
Response.Write("<INPUT TYPE=""TEXT""
NAME=""txtAddress"" VALUE="" & strAddress & """>")
```

٢. بعد أن ينتهى المستخدم من تحديث حقل العنوان يتم إرسال النموذج مرةً أخرى إلى الخادم لمعالجة القيمة الجديدة. ويتم كتابة الكود لاستعادة العنوان من النموذج وتحديث قاعدة البيانات. ومن الضرورى حينئذٍ التأكد من صحة العنوان المدخل.

ولاشك أن هذه الطريقة فى كتابة تطبيقات الويب تعمل بشكل صحيح إلا أن العيب الوحيد يكمن فى تداخل عناصر واجهة المستخدم مع الكود، فلا يوجد ترتيب واضح فى كتابة الكود. أما نماذج الويب على الجانب الآخر، فتحتوى على فصل واضح بين واجهة المستخدم وكود التطبيق. وفيما يلى نوضح كيفية إنشاء نموذج الويب للمستخدم إذا ما أراد تحديث عنوانه:

١. بدلاً من استخدام الحقل INPUT المعتاد لتمثيل حقل العنوان، تتيح لك نماذج الويب استخدام أداة تحكم موجودة على الخادم Server-side Control عبارة عن مربع نص كما يلى:

```
<asp:TextBox id="txtAddress" runat="server">
</asp:TextBox>
```

إذا كانت رموز HTML معتادةً لك، ستجد هناك اختلافاً فى سطر الكود السابق وذلك لأنه لا يمثل كود HTML بالفعل وإنما يمثل أداة تحكم تعمل ناحية الخادم. وإذا قمت باستخدام Visual Studio 2008 فى إنشاء نماذج الويب، فلن تحتاج إلى تعلم أى من صيغ أدوات تحكم الخادم الجديدة حيث تقوم بيئة التطوير بإضافتها نيابةً عنك.

٢. بمجرد تحميل الصفحة، يمكنك تخصيص العنوان إلى الحقل المصاحب من خلال الخاصية **Text** كما يلى:

```
txtAddress.Text = strAddress
```

٣. وأخيراً حينما يقوم المستخدم بنقر زر **Submit**، يمكنك استرجاع العنوان الجديد من أداة التحكم بسهولة شديدة كما يلى:

```
strNewAddress = txtAddress.Text
```

ولعلك ترى معى مدى سهولة الخطوات السابقة والتى تمكنك من العمل مع تطبيقات الويب كما لو أنك تعمل مع تطبيق نوافذى يمكنك من خلاله تعيين الخصائص والأحداث والتحكم فيها.

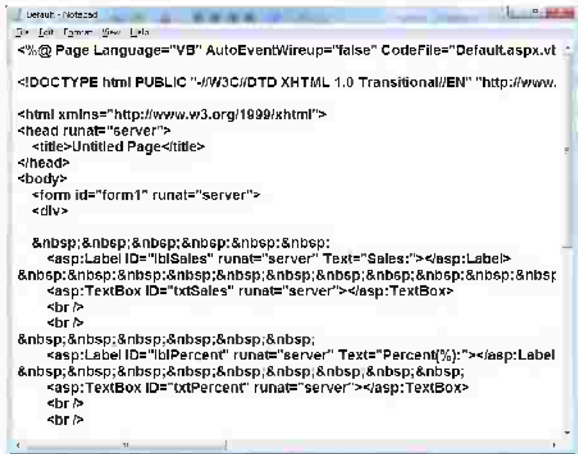
مفهوم أدوات الخادم

تستخدم نماذج الويب أدوات تحكم الخادم بكثرة وهى المكونات التى يتم تنفيذها على خادم الويب والتى تم تصميمها خصيصاً لعزل المصمم عن الطبقة التحتية لنماذج **HTML**. تخيل المثال السابق الذى قمنا فيه باستخدام مربع النص كأحد أدوات تحكم خادم الويب كما يلى:

```
<asp:TextBox id="txtAddress" runat="server">
</asp:TextBox>
```

لاحظ الرموز **<asp:>** و**</asp:>** والتى توضح استخدام أداة تحكم خادم **ASP.NET**. والجدير بالذكر أن أدوات تحكم الخادم تكون مفهومة من قِبَل الخادم فقط ولا يفهمها مستعرض الويب على الإطلاق وإنما يقوم معالج صفحة **ASP.NET** بإرسال كود **HTML** المكافئ إلى المستعرض. لتوضيح ذلك، دقق النظر فى شكل ٣-١التالى الذى يحتوى على كود النموذج داخل **Visual Studio 2008** والكود المكافئ داخل المستعرض حينما يقوم المستخدم باختيار **Source** من قائمة **View** حيث يلاحظ استبدال الرمز **<asp:TextBox>** بالرمز **<INPUT>** وهذا يوضح أن نموذج برمجة نماذج الويب الجديد لا يغير من طريقة عمل المستعرض وإنما يسهل على المطور تطوير تطبيقات الويب المختلفة. فكلما قام المستعرض بطلب صفحة ويب من الخادم، يقوم الخادم على الفور

بإنشاء صفحة HTML المكافئة.

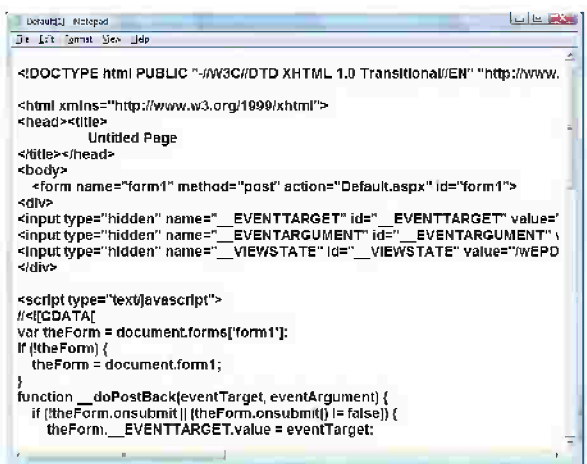


```
<%@ Page Language="VB" AutoEventWireup="false" CodeFile="Default.aspx.vt

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
  <title>Untitled Page</title>
</head>
<body>
  <form id="form1" runat="server">
  <div>

        
     <asp:Label ID="lblSales" runat="server" Text="Sales:"></asp:Label>
           
     <asp:TextBox ID="txtSales" runat="server"></asp:TextBox>
     <br />
     <br />
      
     <asp:Label ID="lblPercent" runat="server" Text="Percent(%):"></asp:Label
          
     <asp:TextBox ID="txtPercent" runat="server"></asp:TextBox>
     <br />
     <br />
```



```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.

<html xmlns="http://www.w3.org/1999/xhtml">
<head><title>
          Untitled Page
</title></head>
<body>
  <form name="form1" method="post" action="Default.aspx" id="form1">
<div>
<input type="hidden" name="__EVENTTARGET" id="__EVENTTARGET" value="
<input type="hidden" name="__EVENTARGUMENT" id="__EVENTARGUMENT" v
<input type="hidden" name="__VIEWSTATE" id="__VIEWSTATE" value="/WEPD
</div>

<script type="text/javascript">
//<![CDATA[
var theForm = document.forms['form1'];
if (!theForm) {
  theForm = document.form1;
}
function __doPostBack(eventTarget, eventArgument) {
  if (!theForm.onsubmit || (theForm.onsubmit() != false)) {
    theForm.__EVENTTARGET.value = eventTarget;
```

شكل ٣-١ يتم ترجمة أدوات تحكم الخادم إلى كود HTML المكافئ

أحداث أدوات التحكم

عند العمل مع أدوات تحكم خادم ASP.NET، يتم كتابة كود إجراءات أحداث هذه الأدوات تماماً كما كنا نفعل فى التطبيق النوافذى. ولكن ليست كل الأحداث المتاحة داخل أداة تحكم Windows تكون موجودة بالضرورة داخل أداة تحكم الخادم المكافئة،

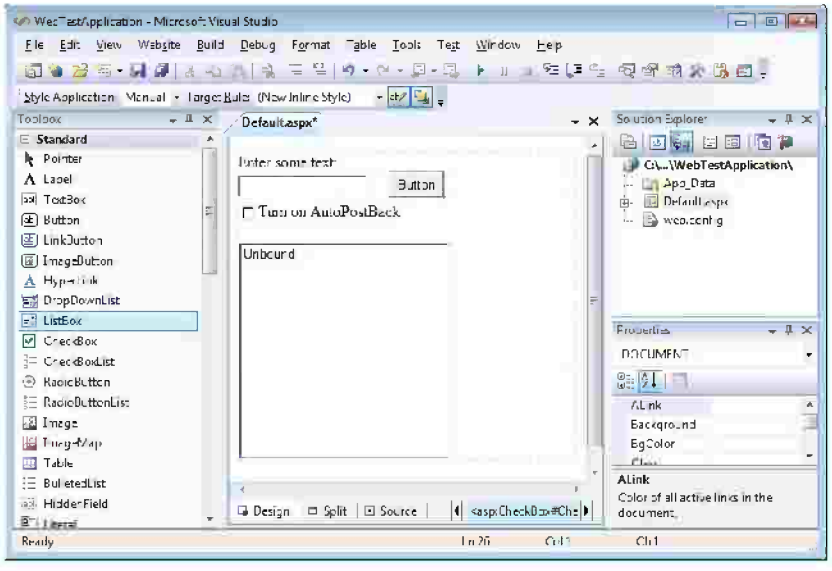
وربما وجد نفس الحدث فى الأداتين إلا أنه يعمل بطريقة مختلفة فى كل منهما. وذلك كله إنما يكون من أجل رفع كفاءة صفحات الويب لأن طريقة المعالجة تختلف بالطبع بين نماذج **Windows** ونماذج الويب. فعلى سبيل المثال، بالنظر إلى الحدث **MouseMove** الموجود مع أدوات تحكم **Windows**، تجد أنه غير عملى بالمرة للاستخدام مع نماذج الويب، لأن فى كل مرة يقوم المستخدم بتحريك مؤشر الفأرة يتم إجراء جولة دائرية بين المستعرض والخادم مما يعمل على بطء عمل الصفحة بشكل ملحوظ وربما لتوقف الاستجابة تماماً، لذا فهذا الحدث غير متاح مع أدوات تحكم الخادم.

وهناك مثال آخر هو الحدث **TextChanged** المصاحب لأداة مربع النص فى نماذج **Windows** والذى يتم استدعاؤه مع كل ضربة مفتاح يقوم المستخدم بأدائها، أما فى **ASP.NET** فيتم استدعاء هذا الحدث مع كل جولة دائرية وفى حالة تغير محتويات مربع النص وذلك لنفس السبب السابق.

يمكنك مع بعض أدوات الخادم التسبب فى المزيد من الجولات الدائرية من خلال الخاصية **AutoPostBack** بتخصيصها بالقيمة **True**. ولتوضيح تأثير هذه الخاصية، تابع معنا الخطوات الآتية:

١. ابدأ بتشغيل بيئة تطوير **Visual Studio 2008** ثم قم بإنشاء موقع ويب جديد.

٢. من مربع الأدوات قم بإضافة أداة عنوان ومربع نص وزر أمر ومربع اختيار ومربع قائمة كما فى شكل ٣-٢.



شكل ٣-٢ إضافة أدوات التحكم إلى النموذج

٣. انقر النموذج نقراً مزدوجاً ثم قم بإضافة الكود التالى للإجراء الحدثى Page_Load:

```
Private Sub Page_Load(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles MyBase.Load
    ListBox1.Items.Add("Page loaded " & Now.ToString)
    If CheckBox1.Checked Then
       TextBox1.AutoPostBack = True
    Else
       TextBox1.AutoPostBack = False
    End If
  End Sub
```

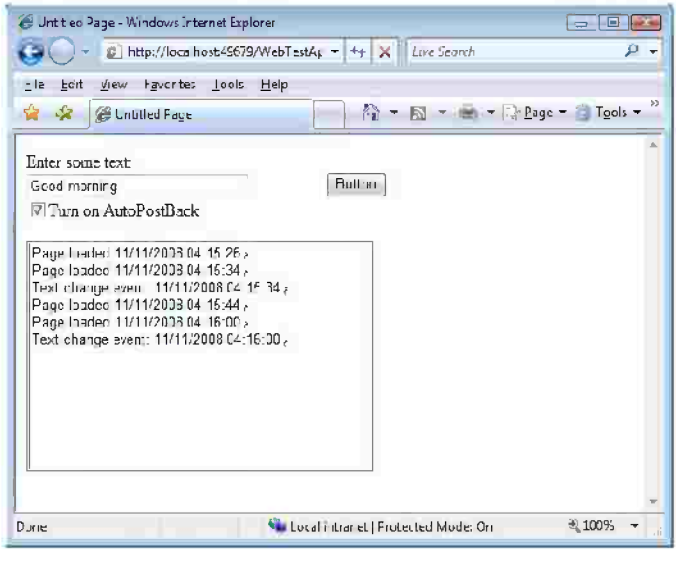
٤. بنفس الطريقة انقر مربع النص نقراً مزدوجاً ثم قم بإدخال الكود التالى داخل الإجراء الحدثى TextChanged:

```
Private Sub TextBox1_TextChanged(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
TextBox1.TextChanged
    ListBox1.Items.Add("Text change event:  " & Now.ToString)
End Sub
```

٥. قم بتشغيل التطبيق ثم قم بإدخال بعض الحروف داخل مربع النص وانقر زر Button لحث المستعرض على إجراء جولة دائرية، تلاحظ تنشيط الحدث TextChanged ودلالة ذلك إضافة النص الموضح بالإجراء إلى مربع السرد.

٦. انقر مربع الاختيار لتنشيطه ثم انقر زر Button مرةً أخرى وبذلك تكون قد خصصت القيمة True للخاصية AutoPostBack.

٧. قم بإدخال نص جديد إلى مربع النص ثم اضغط مفتاح Tab من لوحة المفاتيح للخروج من حقل مربع النص، تلاحظ تنشيط الحدث TextChanged دون الحاجة إلى نقر زر Button (انظر شكل ٣-٣).



شكل ٣-٣ نتيجة تنشيط أو تعطيل الخاصية AutoPostBack

فتبعاً لقيمة الخاصية AutoPostBack، يتم تنشيط الحدث TextChanged إما بنقر زر الأمر أو بفقد مربع النص للتركيز. وفى كلتا الحالتين يعكس الحدث التغييرات التى حدثت للأداة منذ آخر جولة، وبعبارة أخرى إذا احتوى مربع النص على كلمة Welcome ثم قمت بكتابة العديد من الكلمات ثم استبدلتها مرةً أخرى بكلمة Welcome فهذا يعنى عدم وجود أى تغيير بمربع النص.

كتابة كود العميل Client-side Script

يتضمن النموذج البرمجى لنماذج الويب تنفيذ الكود الموجود جهة الخادم Server-side code فلا يوجد أى عمليات معالجة مع الصفحة الموجهة من الخادم إلى العميل وهـذا فى الواقع أحد نقاط تميز تطبيقات الويب لأنك تحتاج إلى القليل من الجهد كى يعمل تطبيقك لدى العميل. ولكن على الرغم من ذلك فإن معظم مستعرضات الويب تدعم استخدام كود العميل أو Client Scripting مثل لغة JavaScript أو لغة VBScript. والملاحظ هو انتشار JavaScript بكثرة عن VBScript لأنها تعمل مع مستعرضى الويب الشهيرين Internet Explorer وNetscape Navigator.

وعلى الرغم من أنك لست فى حاجة ماسة إلى كتابة كود العميل فى تطبيقات الويب، إلا أن مصممى الويب كثيراً ما يقومون بإضافة مثل هذا الكود لتنفيذ بعض المهام داخل مستعرض المستخدم لأحد الأسباب التالية:

- التقاط الأحداث التى لا يستطيع الخادم التقاطها مثل حركة المؤشر داخل الصفحة.
- تعيين واجهة مستخدم متحركة أو ديناميكية عن طريق تظليل أو تحريك أجزاء من صفحة الويب.
- إنشاء نوافذ وليدة من المستعرض لإظهار مجموعة من صفحات الويب فى وقتٍ واحد.

مثال تطبيقى

سنقوم فى الجزء المتبقى من هذا الفصل بتوضيح مثال تطبيقى على كيفية إنشاء تطبيق الويب من داخل بيئة تطوير Visual Studio 2008. تابع معنا الخطوات الآتية:

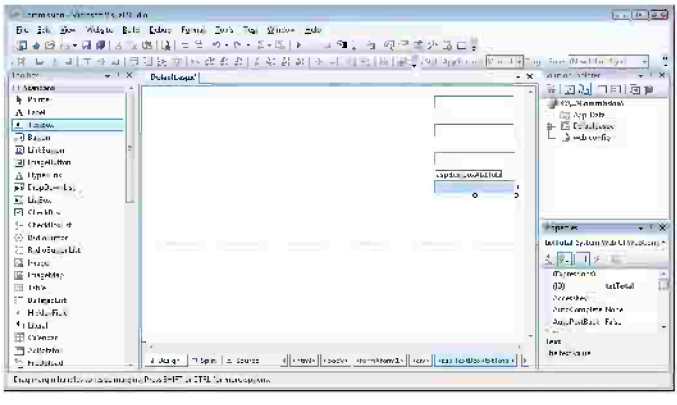
١. من بيئة تطوير Visual Studio 2008، افتح قائمة File من شريط القوائم ثم اختر New Web Site من القائمة المنسدلة، يظهر المربع الحوارى المعتاد New Web Site.

٢. اختر الرمز **ASP.NET Web Site** من المربع **Templates** الموجود بالجزء العلوى.

٣. قم بتعيين اسم مناسب للمشروع وليكن **Commission** وقم أيضاً بتحديد مكان المشروع وذلك من خلال القائمة المنسدلة **Location** والقائمة المجاورة.

٤. تأكد من اختيار **Visual Basic** بالقائمة **Language** ثم انقر زر **Ok** لإغلاق المربع الحوارى وإنشاء الموقع.

٥. أضف إلى نموذج النافذة أربعة مربعات نصوص بأسماء **txtSales** و **txtPercent** و **txtSalary** و **txtTotal** مع وضعها جهة اليمين ومحاذاتها (انظر شكل ٣-٤).



شكل ٣-٤ نموذج النافذة بعد إضافة مربعات النصوص الأربعة

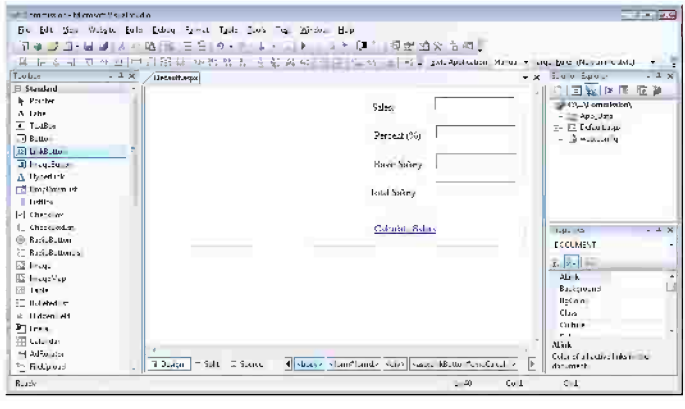
٦. الآن، وبعد أن رسمنا مربعات النصوص ينبغي أن نضع عنواناً لكل مربع حتى يدرك المستخدم المحتويات المطلوبة في كلٍ منهم. قم بإضافة العناوين الأربعة بجوار مربعات النصوص المناظرة مع اختلاف الاسم والتعليق الخاص بكل منها. يحتوى الجدول ٣-١ التالى على خصائص العناوين الأربع التي سننشئها.

جدول ٣-١ خصائص العناوين التي توضح مدخلات التطبيق

الاسم Name	العنوان Text
lblSales	Sales :
lblPercent	Percent(%) :
lblBasic	Basic salary :

الاسم Name	العنوان Text
lblTotal	Total Salary :

٧. آخر ما نحتاج من العناصر هو زر الارتباط الذى يتشابه فى طريقة عمله كثيراً مع زر الأمر المستخدم فى النماذج النوافذية، وهو مسئول عن بدء عملية ما. ابدأ بالنقر على الأداة LinkButton في مربع الأدوات لاختيارها ثم ارسم الزر على النافذة وامنحه اسما مميزا وليكن **cmdCalculate** كما فعلت بالعناصر السابقة. أيضا للزر خاصية **Text** مثل العنوان وتحوي الكلمات التي ستظهر على سطح الزر وغالبا ما تعبر عما يقوم به التطبيق عند الضغط على الزر، ضع التعليق **Calculate Salary** في خاصية **Text** للزر (يجب أن يظهر النموذج الآن كما فى شكل ٣-٥).



شكل ٣-٥ نموذج النافذة بعد إضافة زر الارتباط

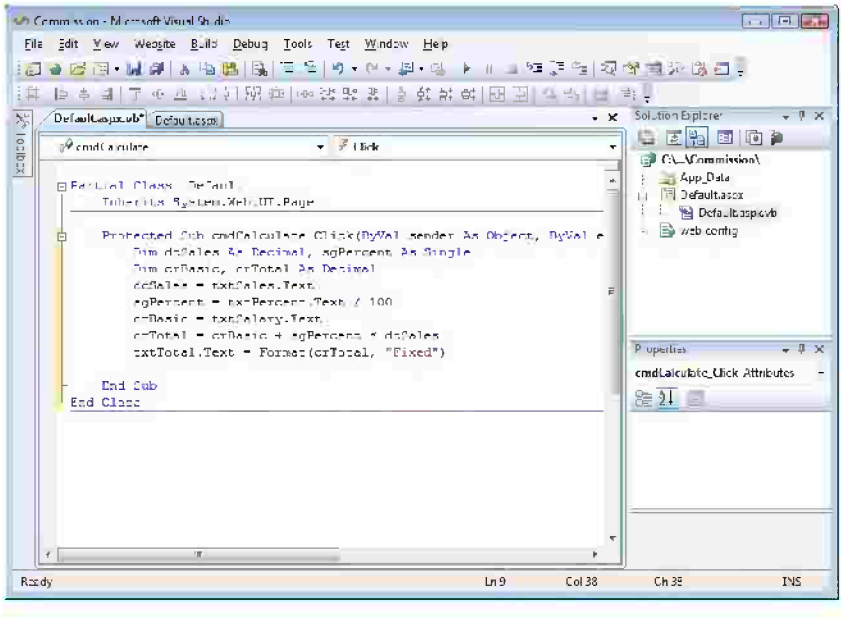
من الأفضل الآن فى هذه المرحلة أن تقوم بحفظ عملك تحسباً لانقطاع التيار أو حدوث خطأ ما. لأداء ذلك، افتح قائمة **File** من شريط القوائم ثم اختر **Save All** من القائمة المنسدلة أو اضغط الزر مباشرةً من شريط الأدوات.

٨. قم بكتابة الكود التالى داخل إجراء حدث نقر الزر **cmdCalculate**

```
Dim dcSales As Decimal, sgPercent As Single
```

```
Dim crBasic , crTotal As Decimal
dcSales = txtSales.Text
sgPercent = txtPercent.Text /100
crBasic = txtSalary.Text
crTotal = crBasic + sgPercent * dcSales
txtTotal.Text = Format(crTotal,"Fixed")
```

يجب أن تظهر نافذة الكود الآن كما فى شكل (٣-٦).

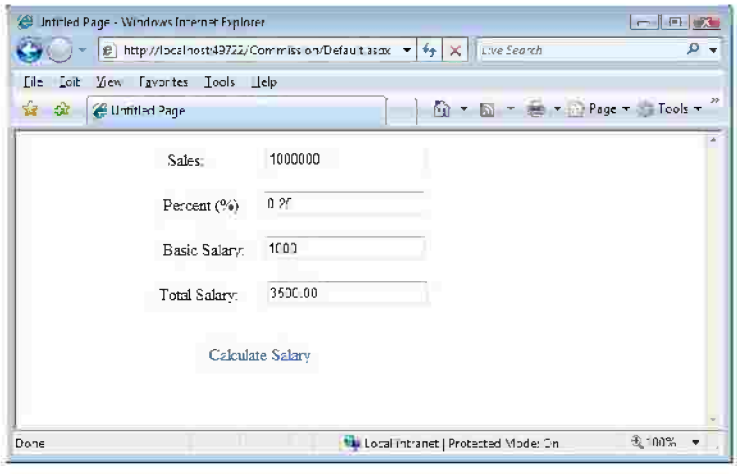


شكل ٣-٦ نافذة الكود عند كتابة إجراء الحسابات

لتشغيل التطبيق اضغط زر ▶ من شريط الأدوات، يتم تشغيل التطبيق ويظهر النمـــوذج داخل مستعرض الويب المثبت على حاسبك. للاختبار أدخل القيم الآتية في المربعات:

Sales	**1000000**
Percent	**0.25**
B. Salary	**1000**

ثم اضغط الارتباط **Calculate Salary**. ينبغـــي أن تكـــون قيمـــة **Total Salary** المعروضة مساوية ٣٥٠٠ (انظر شكل ٣-٧). بعد اختبار التطبيق بعدة قيم يمكنك العودة لبيئة التطوير بنقر زر إغلاق مستعرض الويب بالركن الأيمن العلوى.



شكل ٣-٧ نموذج التطبيق داخل مستعرض الويب